# Reassessing Caching Performance in Information-Centric IoT

Jakob Pfender, *Student Member, IEEE,* Alvin Valera, *Member, IEEE,*
and Winston K. G. Seah, *Senior Member, IEEE*

***Abstract*—The performance of in-network caches has traditionally been evaluated using well-understood metrics such as the cache hit rate. This extends to the field of Information-Centric Networking (ICN), where caching strategies are evaluated in the same way. In this paper, we argue that in the subdomain of information-centric Internet of Things (IoT), these traditional metrics are not sufficient to describe caching performance. Using a series of experiments on real hardware as a demonstration, we show the shortcomings of the cache hit rate metric and introduce the two new metrics *cache access factor* and *cache latency factor* that provide deeper insights into the effectiveness of in-network caching strategies.**

***Index Terms*—Information-Centric Networking, Named Data Networking, Internet of Things, In-Network Caching, Network Topology.**

## I. Introduction

There is a growing mismatch between the original Internet design and its current use. The Internet and its underlying TCP/IP protocol suite was originally designed for host-to-host communications whereas modern applications use it for retrieving and disseminating content without regard to the physical source of the information. This mismatch is posing significant difficulties on the underlying Internet architecture [1]. The content-centric nature of modern applications and application domains such as the Internet of Things (IoT) is leading to the emergence of new messaging paradigms.

One such promising paradigm is Information-Centric Networking (ICN). Its content-centric nature and slim network stack make it an ideal candidate for a future network architecture for IoT applications [2]–[5]. One of the core tenets of ICN is *in-network caching*, whereby nodes maintain caches for storing transit content. This ensures that relevant content is readily available across the network, even if the original producer is not reachable.

The question of where in the network should content be cached is one of the most defining problems of in-network caching research in ICN [6]–[10]. This question is especially relevant in the domain of information-centric IoT, where hardware is typically limited, which places more importance on the effective use of available memory. While caching approaches for information-centric IoT have been studied, much of the existing research still approaches the domain with preconceptions carried over from the fields of mainstream ICN or even traditional networking, without fully taking the idiosyncrasies of IoT into account. As such, many proposed strategies are overly complex, where preliminary research has indicated that simpler, low-overhead strategies may be preferable [11], [12]. Furthermore, even if strategies are developed specifically with deployment in the IoT in mind, they are often still evaluated using metrics from traditional networking, without first investigating whether these metrics are useful for analysis in the IoT space [13]–[21].

One such metric is the *cache hit rate*. A ubiquitous metric in any domain where caches are prevalent, the cache hit rate describes the percentage of requests that are satisfied by a cached copy of the requested content rather than by the original producer of the content [22], [23]. In ICN, this metric is generally very useful in evaluating the performance of a given caching strategy, as the goal of in-network caching is to allow content to be retrieved rapidly and reliably. A high cache hit rate indicates that the caching strategy is effective at distributing relevant content across the network, because a cache hit indicates that the request in question was able to be satisfied by a node closer to the consumer rather than the original producer. This implies both a lower content delivery latency as well as reduced load on producers, both desirable qualities in any network.

However, care must be taken when using the cache hit rate as the primary indicator of caching effectiveness. The cache hit rate metric has its origins in traditional computing, where there are typically several levels of Central Processing Unit (CPU) caches that differ tremendously in access latency [24]. In this context, cache misses for a single request incur latency penalties that increase by orders of magnitude with each cache miss on subsequent cache levels. Increasing the cache hit rate is thus synonymous with a clear increase in performance. This mentality has carried over into the evaluation of in-network caching strategies in traditional networking, and subsequently into ICN.

This paper will therefore challenge the well-entrenched use of traditional ICN metrics particularly cache hit rate in the context of IoT. Using an experiment study on physical IoT hardware, this paper will demonstrate that cache hit rate alone is not a sufficient metric to characterise caching performance, and subsequently introduce two new metrics, the *cache access factor* and the *cache latency factor*, that take into account both cache hits and the reduction in hits/latency to provide a more nuanced understanding. The rest of the paper is organised as follows: Section II provides a concise overview of ICN and caching, followed by Section III that motivates and proposes the new metrics. Section IV and Section V discuss the experimental validation to support the proposed metrics before concluding in Section VI.

J. Pfender, A. Valera, and W. Seah are with the School of Engineering and Computer Science, Victoria University of Wellington, New Zealand. E-mail: {jakob.pfender, alvin.valera, winston.seah}@ecs.vuw.ac.nz

## II. Information-Centric Networking and Caching Performance

*Content-Centric Networking (CCN)*, was first proposed by Jacobson *et al*. in 2009 [25]. CCN and its successors are often grouped together with similar approaches under the umbrella term *Information-Centric Networking (ICN)*. Their main contribution constitutes a complete overhaul of the existing approach to networking, replacing the host-based addressing system of IP with a new system that treats named content objects as first class network entities. CCN decouples location from identity by basing its routing logic entirely on the unique names of the routed content objects instead of unique addresses of hosts. This allows network participants to be agnostic about where their requested content actually resides in the network, as well as enabling transparent systems for in-network caching and replication of data, thus increasing availability and performance.

### A. Named Data Networking

*Named Data Networking (NDN)* [26], which has emerged as one of the most popular ICN solutions in recent years, is a comprehensive implementation and extension of the ideas brought forward by CCN. While it is not the only ICN implementation under active development, research suggests that it is the most suited for IoT applications thanks to its scalable naming technique and flexible approach to caching [27].

NDN defines two fundamental types of packets used for communication: *Interest* and *Data*. When a consumer wants to request content, it puts the unique name of that content into an Interest packet and sends it to the network. Routers in the network then forward the Interest towards the named data's producer based on the route entries in the Forwarding Information Base (FIB). For an Interest that a router has forwarded but not fulfilled yet, it will add an entry in the Pending Interest Table (PIT) and note the requesting node; subsequent requests (Interest packets) for the same content from other nodes will not be forwarded but updated in the PIT. Once the Interest reaches a network node that has the named data, a Data packet carrying the data is returned to the consumer(s). Every router that forwards a Data packet will keep a copy in its Content Store (CS) so that it can respond to future Interest packets requesting for the same content.

Research on how to apply ICN paradigms to Wireless Sensor Networks (WSNs) and the IoT is still a comparatively young field. However, researchers are laying the foundations of what is sometimes called the *Named Data Network of Things (NDNoT)* [28] — in particular, current research is focused on how to adapt and optimise existing ICN strategies to the unique environment of IoT, with its unreliable links and devices that are constrained in both memory and processing power.

### B. Performance Metrics

In the existing literature, there is a wide range of metrics that have been used to evaluate the performance of caching strategies. In this section, we present two of the most widely used metrics, viz., *cache hit rate* and *content delivery latency*.

*1) Cache Hit Rate:* Cache hit rate is the performance metric that is often used to characterise the performance of in-networking caching[1]. It is essentially the ratio of content objects that are retrieved as a cached copy from another node in the network as opposed to being retrieved from the original producer. The cache hit ratio $R_{CH}$ is defined as:

$$R_{CH} = \frac{C_{cache}}{C}, \tag{1}$$

where $C$ is the total number of content objects retrieved and $C_{cache}$ is the number of content objects retrieved from the cache of an intermediate node that is not the content producer.

In general, a higher cache hit rate is desirable, as it means that (*i*) content delivery times are reduced as content requests are being fulfilled without having to traverse the full path to the producer and (*ii*) strain on individual producers is reduced as the number of requests routed to them goes down, thus increasing battery life and reducing the probability of dropped packets due to saturated buffers.

*2) Content Delivery Latency:* The *content delivery latency* measures the time elapsed from the instant an Interest is generated (*interest generation*) by a consumer node up till the time it is satisfied (*interest satisfaction*) by the receipt of the requested content, including possible retransmissions [29]. Delivery latency may be affected by several caching-independent factors, such as network congestion and density, but it is also affected by cache diversity — the better the average availability of content chunks across the network, the lower the delivery latency. If we assume that congestion and density stay roughly the same, we can use the delivery latency to evaluate a caching strategy's effectiveness in making content quickly available.

### C. Related Work

There have been a number of studies on the performance of ICN caching strategies, both in traditional networking as well as in the IoT space. Few, if any, proposed metrics that can better model performance in the IoT domain.

Caching in traditional ICN has been studied extensively, both in terms of performance in general [13]–[17] and content delivery latency in particular [18]–[21]. While these contributions are extremely valuable, it is not a given that the results can be applied directly to IoT environments since they do not take that domain's idiosyncrasies into account. The most comprehensive surveys of caching schemes specifically for information-centric IoT were presented by Arshad *et al*. [27], [30] and Gupta *et al*. [31]. These papers, however, are pure surveys, with no experimental evaluation or comparison of the presented strategies.

The first comparative studies on ICN caching strategies specifically in the IoT were carried out by Hail *et al*. [22] and Meddeb *et al.* [23], both of whom use simulated environments for their evaluations. Hail *et al*. compared the *Cache Everything Everywhere (CEE)* and probabilistic (with $p = 0.5$) caching decision strategies. Their results only considered the

[1] A *cache hit* occurs whenever an Interest is served by a cache in the network instead of the requested content's original producer.

basic performance metrics of cache hit ratio, data retrieval delay, and Interest retransmissions. Meddeb *et al*. compared *CEE*, *Leave Copy Down (LCD)*, *ProbCache*, *Betw* [7], *Edge Caching* [32], and their own *Consumer-Cache* strategy and evaluated them in regard to cache hit ratio, number of evictions, hop reduction ratio, and data retrieval delay.

The first studies using physical IoT hardware operating in realistic conditions were presented in our previous work [11], [33], where the strategies presented in this paper were evaluated using the same basic metrics as existing literature.

## III. New Performance Metrics for Caching

The use of cache hit rate for characterising the performance of caching strategies may be justifiable in the Internet but in the context of IoT which relies on wireless sensor networks, its suitability needs to be examined. In this section, we will put forth arguments on why cache hit rate is not an appropriate metric for information-centric IoT and introduce new performance metrics that take into account the unique characteristics of information-centric IoT.

### A. Pitfalls of Cache Hit Rate in IoT

Delivery paths in the Internet are long, with many hops along the way. The distance between two adjacent hops is theoretically unbounded. Thus, a cache miss can similarly be interpreted as incurring a potentially significant latency penalty. However, these conditions are not true in the wireless deployments found in the IoT. The distance between two adjacent hops in a wireless network has a clear upper bound defined by the device's transmission range. A typical IoT deployment features relatively evenly-spaced nodes in order to minimise costs while ensuring reliable transmissions. **This means that the latency penalty incurred by cache misses scales linearly instead of exponentially, reducing their negative impact.** Therefore, instead of aiming to simply maximise the number of cache hits regardless of where on the delivery path the hit occurs, we argue that in IoT, it is more important to maximise *cache access*, i.e. to make sure that caches are optimally placed in the network in order to minimise content delivery latency.

### B. Hop Count and Latency Reduction

As a precursor to latency reduction, we introduce the *hop count reduction* metric. For each Interest, the number of hops between its origin and the owner of the prefix it is requesting is denoted as the *distance to source*. In other words, this is the number of hops the Interest/Data packet would always have to travel if there were no caches in the network. This distance can then be compared to the actual number of hops taken by the Data packet on the way back. This measure is called *hops to hit* as it denotes the number of hops it actually took for the Interest to reach a cache hit. The more efficient a caching strategy, the more content will be available in a cache closer to the consumer, leading to a lower average hops to hit value. The difference between the distance to source and the hops to hit is denoted as the *hop count reduction*, and the *hop reduction ratio* is the ratio between the hop count reduction and the distance to source. For a single content delivery operation $c$, the hop reduction ratio is thus defined as:

$$HRR_c = \frac{to\_source_c - to\_hit_c}{to\_source_c}, \tag{2}$$

where $to\_source_c$ is the *distance to source* between the prefix owner of $c$ and the consumer that requested it and $to\_hit_c$ is the *hops to hit* the content chunk $c$ in a cache. Thus, the more hops a delivery path is reduced by (i.e. the lower $to\_hit_c$), the higher $HRR_c$ for that content delivery operation.

Analogous to the hop reduction ratio $HRR_c$, we can measure the *latency reduction ratio* $LRR_c$. It is defined as:

$$LRR_c = \frac{expected\_latency_c - actual\_latency_c}{expected\_latency_c}, \tag{3}$$

where $expected\_latency_c$ is the content delivery latency that would be expected for an operation $c$ if there were no intermediate caches (i.e. if the content had to be retrieved from the original producer.

### C. Cache Access Factor and Cache Latency Factor

Cache hit rate only paints a partial picture of content accessibility in the network. Its main drawback is that it contains no information about how accessible for consumers the caches actually are. For example, a caching strategy that keeps contents close to the core might in theory have a very high cache hit ratio while not significantly reducing path lengths. The hop and latency reduction ratio provide more detailed understanding of this behaviour, but can be quite complex if the caching strategy behaves differently depending on path length. A more comprehensive approach to quantifying performance is required. To that end, we will now introduce two new metrics — *cache access factor* and *cache latency factor* — that aim to better model caching performance.

*1) Cache Access Factor:* The cache access factor $F_{CA}$ takes both cache hit and hop reduction ratio into account to produce a single metric that weights pure content accessibility with the average reduction of delivery paths. The cache access factor is defined as:

$$F_{CA} = R_{CH} \cdot \frac{\sum_{i=0}^{n} HRR_i}{n}, \tag{4}$$

where $R_{CH}$ is the cache hit rate, $n$ is the number of content objects delivered, and $HRR_i$ is the *hop reduction ratio* for a content object $i$ as defined in Equation (2).

*2) Cache Latency Factor:* Analogous to the cache access factor $F_{CA}$, which combines cache hit rate and hop reduction ratio, we can also examine the cache hit rate in conjunction with the reduction in content delivery latency. This lets us derive the *cache latency factor* $F_{CL}$, which is defined as follows:

$$F_{CL} = R_{CH} \cdot \frac{\sum_{i=0}^{n} LRR_i}{n}, \tag{5}$$

where $R_{CH}$ is the cache hit rate, $n$ is the number of content objects delivered, and $LRR_i$ is the *latency reduction ratio* for a content object $i$ as defined in Equation (3).

In effect, $F_{CA}$ and $F_{CL}$ condense the expected gains in cache utilisation of a given caching strategy into more accessible terms. In a deployment without caching, $F_{CA}$ and $F_{CL}$ are 0 because $R_{CH}$ is 0. Maximising $F_{CA}$ and $F_{CL}$ requires both maximising $R_{CH}$ as well as minimising the hops to hit and the latency (or rather, maximising the latency reduction). A high cache hit rate alone does not suffice if path lengths and therefore latency are not significantly reduced.

## IV. Experiment Description

This section presents an experiment designed to demonstrate the latency effects of different caching strategies and the metrics that can be used to characterise them. We start by describing the results using common metrics, then show why these metrics are insufficient for a complete analysis, and then introduce advanced metrics to address these shortcomings.

### A. Experiment Setup

The experiments for this study were run on the **FIT IoT-LAB** [34] open testbed. The IoT hardware used is IoT-LAB's specially developed **M3 node**[2], which has an STM32 (ARM Cortex M3) microcontroller with 512 kB ROM and 64 kB RAM and an Atmel AT86RF231 [35] 2.4 GHz transceiver operating on IEEE 802.15.4 [36]. The firmware for the nodes is a simple **RIOT-OS** [37], [38] application using **CCN-lite**[3] as the ICN implementation, modified to support the different caching strategies.

The experiments were conducted on the *Grenoble* site[4] of the IoT-LAB testbed. The site features more than 380 M3 nodes, which are distributed across the rooms and corridors of one floor of an office building (see Fig. 1). This means that nodes are subject to realistic conditions found in indoor IoT deployments, such as multipath effects, reflection, and absorption caused by walls, doors, and windows made of various materials, as well as unpredictable interference by other wireless signals and people moving around the building. These conditions mean that the behaviour of the network is very close to what might be expected in a real-world deployment.

Of the 380 available nodes, each experiment run is conducted on an arbitrary subset of 50 nodes (chosen randomly each time), each of which act as producers, consumers, and relays at the same time. This ensures that the logical topology is different in each experiment run and also that the nodes will not be too strongly connected due to having a large number of one-hop neighbours. This is desirable as it allows us to study the effects of unreliable connections more closely. The transmission range of individual nodes is not enough to reach all other nodes in the building, so communication will be predominantly multihop. In a typical topology generated by this random selection of nodes, the mean path length is between 3 and 4 hops and the maximum is 11 hops. This kind of multihop setup is commonly found in the industrial monitoring domain. Real-world deployments tend to have slightly longer average paths, but this scale is infeasible to achieve within the constraints of physical testbeds such as IoT-LAB.

For this study, cache sizes are kept intentionally small. Each node's cache can store up to 5 unique content chunks (all content chunks have the same size). This small cache size was chosen for two reasons. For one, RAM is extremely limited in IoT devices. The M3 nodes used in this study have 64 kB of RAM. A constant fraction of this RAM is occupied by the operating system (4.4 kB) and the CCN-lite network stack (8.7 kB) [39], leaving about 50 kB that have to be shared between the CCN-lite heap (comprising CS, FIB, and PIT), and the actual application running on top of the network stack. However, these numbers are at the upper end of typical RAM sizes for class 2 devices. Class 1 devices with RAM on the order of 10 kB [40] also need to be considered. In these devices, the OS and network stack already need to be pruned for features, and the remaining CCN-lite heap size will be at most 1 kB [41]. Depending on the nature of the data transmitted by the application, available cache space may thus be severely limited. This motivates the decision to limit the number of CS entries in this way in order to be able to assess expected performance under these conditions.

The secondary motivation for limiting the number of CS items to 5 is that many adverse effects of ICN content availability could simply be countered by over-provisioning, i.e. providing more cache space (if the available RAM allows), thus ensuring content distribution. This means that performance differences between caching strategies become less pronounced as cache size increases. Therefore, it is more interesting to look at performance under limited cache sizes, since this is where differences will be most noticeable. The *Least Recently Used (LRU)* cache replacement policy is used in all experiments. As noted in previous work [11], the choice of cache replacement policy has little to no impact on the performance of in-network caching.

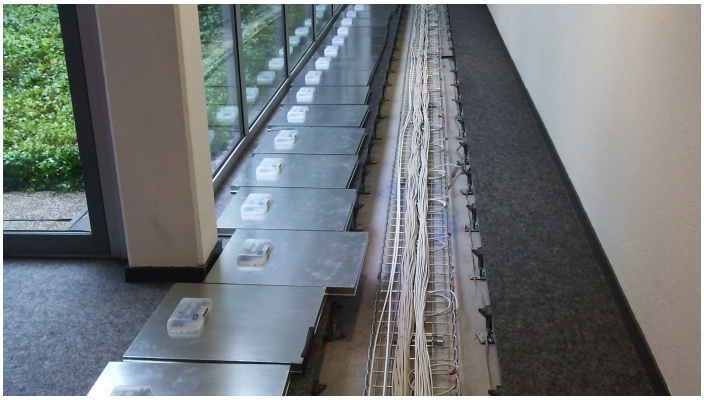
The experiments are managed by a control script using the IoT-LAB API, which provides full control over all node serial interfaces. All nodes will periodically request contents with random IDs in $\{0, \ldots, 49\}$ from each of the prefixes in its FIB. Interest and Data packets are handled as specified by the NDN standard. The first time a node receives an Interest for a content chunk it owns, it produces that content chunk (the actual payload is irrelevant for this study) and sends it back towards the consumer. Caching of content chunks at intermediate nodes is dictated by the caching strategy selected for the study.

### B. Caching Strategies

The caching decision strategies examined in this study are *CEE* [25], *LCD* [42], *Prob(p)* [29], *ProbCache*/*ProbCache-Inv* [9], *Approximate Betweenness Centrality (ABC)* [43], *Labels* [44], and *Intervals* [45].

*CEE*, also known as *Leave Copy Everywhere (LCE)*, is the most straightforward caching decision strategy that is used in traditional ICN [25], [26]. Nodes will attempt to cache every incoming content chunk that is not already in their CS. If

[2]https://github.com/iot-lab/iot-lab/wiki/Hardware_M3-node
[3]https://github.com/cn-uofbasel/ccn-lite
[4]https://www.iot-lab.info/deployment/grenoble/



(a) Deployment of nodes in the IoT-LAB Grenoble site

(b) IoT-LAB nodes in the Grenoble Senslab space

Fig. 1. The Grenoble site of the IoT-LAB testbed was used for experimental evaluation

caching everything at every node is not an option, but the caching process is to remain simple, *LCD* [42] is a viable option. In *LCD*, content is always cached only at the next hop from the node where the cache hit occurred, i.e. initially one hop downstream from the producer, and one hop further downstream with each subsequent request.

To increase cache diversity and decrease redundancy, the easiest solution is to simply introduce a certain probability that any given content chunk will not be cached, as proposed by *Prob(p)* [29]. It simply sets an *a priori* probability $p$ that a given node will store a given content chunk. Upon receipt of a new content chunk, the node generates a random number between 0 and 1. If the generated number is smaller than $p$, the content is stored in the cache; otherwise, it is forwarded without being cached.

Instead of defining an *a priori* caching probability that is the same for every node and also to take into consideration the network topology, *ProbCache* [9] computes the caching probability of a given content chunk based on the distance between producer and consumer as well as the location of the caching node on the path, essentially caching content further away from the producer. *ProbCache-Inv* is identical to *ProbCache* in every way except the final caching probability is inverted to cache closer to the producer.

Centrality-based caching strategies are a sub-family of the topology-based approaches. Betweenness centrality describes the number of times a given node lies on one of the paths between all pairs of nodes in the network and has been found to be a useful indicator of node importance in a network [46]. Unfortunately, in terms of implementation, centrality-based approaches require a costly setup phase before they can begin operation, and if the topology is dynamic — e.g. with mobile participants — these network-wide calculations have to be repeated periodically, leading to significant overhead. *ABC* [43] overcomes this by estimating a node's betweenness centrality value based on the number of Interest/Data packets that pass through the node.

When caching decisions take take more than local information into account, they are deemed to be cooperative. We selected implicit cooperation schemes in our study as they incur lower communication overheads, without the need for nodes to exchange information among themselves. In *Labels* [44], each node is assigned a fixed label $l < k$ (at setup time) and only caches content chunks whose IDs modulo $k$ are equal to $l$. This ensures that cached content is automatically stratified into equal subsets and evenly distributed across the network without the overhead of explicit coordination between nodes. Another implicitly coordinated caching strategy, *Intervals* [45], uses hop distance to determine the caching decision. Data packets are extended by a pre-determined *data interval* value $i$. Each node along the path decrements this value by 1 when forwarding the packet. If a node decrements its value to 0, the packet is cached at that node and the interval is reset to $i$.

### C. Experiment Topology

In ICN, the logical topology is a direct result of the forwarding paths stored in the nodes' FIBs. The FIBs codify how Interests are forwarded and thus how content is distributed across the network. Therefore, getting a sense of a network's logical topology requires knowledge about how its FIBs are constructed. There is no universal answer to this, because ICN enforces no standards for how FIBs are populated. However, in most cases, the contents of the FIBs are the direct result of the routing algorithm that is used by the producers to advertise their content. The way in which nodes learn about their neighbours' contents and in turn inform their own neighbours will dictate what their FIBs will look like. Ultimately, this means that the routing algorithm dictates the entire network's logical topology.

In this paper, we introduce some nomenclature useful for discussing topology types. In general, any IoT network topology can be placed somewhere on a scale between two extremes: the *core* and *edge* topologies. A core topology (Fig. 2) is defined by the paths between the producer and the consumers intersecting nearer to the producer (the "core"); each path has only one consumer attached to it at the edge. In such a topology, the ideal caching location would be close to the producer (what Wang *et al*. [47] call a *Type III* caching strategy), as this would allow us to alleviate strain on the producer while serving the maximum number of consumers with cached copies of the data. Conversely, an edge topology

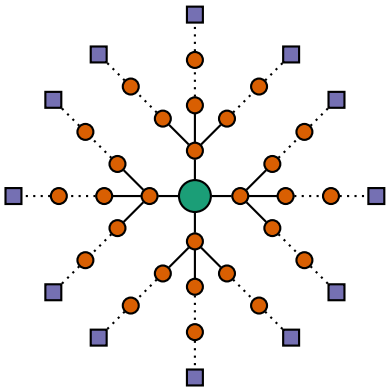


Fig. 2. Core topology

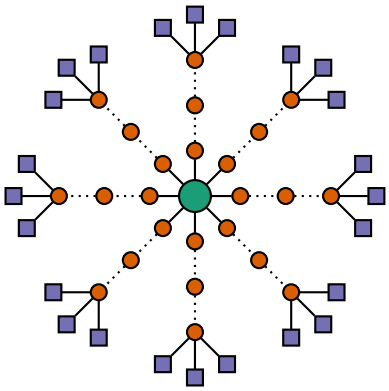


Fig. 3. Edge topology

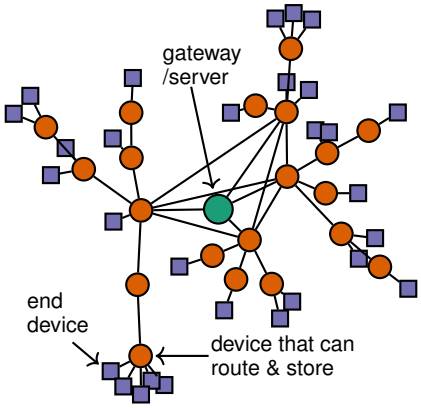



Fig. 4. Integrated (hybrid) topology

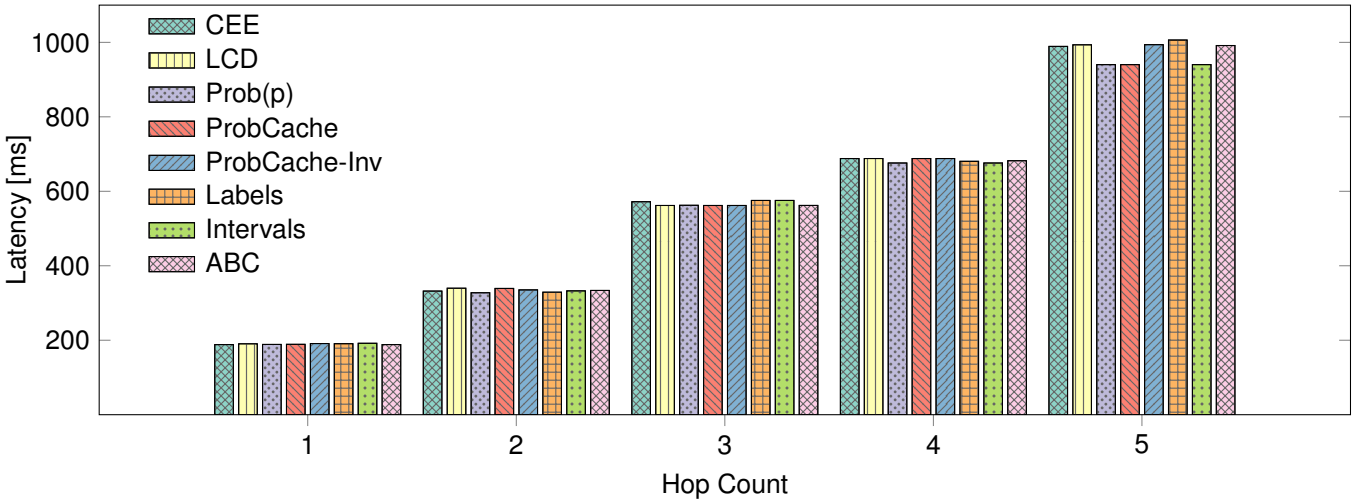



Fig. 5. Relation between hop count and latency for different caching decision strategies

(Fig. 3) is defined as having multiple individual paths from the producer out towards the consumers (the "edge"), which intersect further from the core. In this topology, it would be more beneficial to cache closer to the edge nodes where paths intersect (a *Type II* caching strategy [47]), as this would reduce the need for requests to be routed all the way to the core. They differ in whether content delivery paths are more likely to intersect, near the core or near the edge, and caching strategies that take this effect into account can perform more consistently across all topology types, while others may perform strongly in one topology but fall behind in others.

As the pure core and edge topologies are idealised examples unlikely to be encountered in this form in real deployment scenarios, we also utilise a more realistic topology as shown in Fig. 4, which features elements of both core and edge topologies (although paths tend to intersect closer to the core, placing it more in the former category.) Such hybrid heterogeneous network topologies are increasing common in edge/fog computing scenarios where IoT devices with routing/storage capabilities (orange circles) relay data between the IoT end devices (purple squares) and a gateway/server (large green circle in the middle) [48], [49].

In comparison, *CEE* [25] and *LCD* [42] used linear network topologies where nodes are either directly connected or have one to two intermediate routers. *Intervals* [45] simulated a random mesh network where nodes route via a shortest path to a destination giving a topology that tends towards the edge topology. *Prob(p)* [29] also used a random wireless mesh network with controlled flooding that would tend towards the hybrid topology. Similarly, *Labels* [44] validated their approach using a hybrid topology based on the European backbone network. Lastly, *ProbCache*/*ProbCache-Inv* [9] used a binary tree topology and *ABC* [43] used both core and edge topologies.

The experiments presented in this study were conducted on all three topology types, viz. core, edge and hybrid, and for most metrics discussed here, results are shown separately for all types. Each experiment for a specific network topology and set of parameters was executed multiple times, results averaged and checked to ensure that we achieved a stable representation of the performance [50]. For metrics where there is no discernible difference between topology types, the results from the integrated topology are shown.

## V. Evaluation

### A. Relation Between Hop Count and Latency

Fig. 5 shows how the latency (specifically, the content delivery latency, which is the time between Interest generation

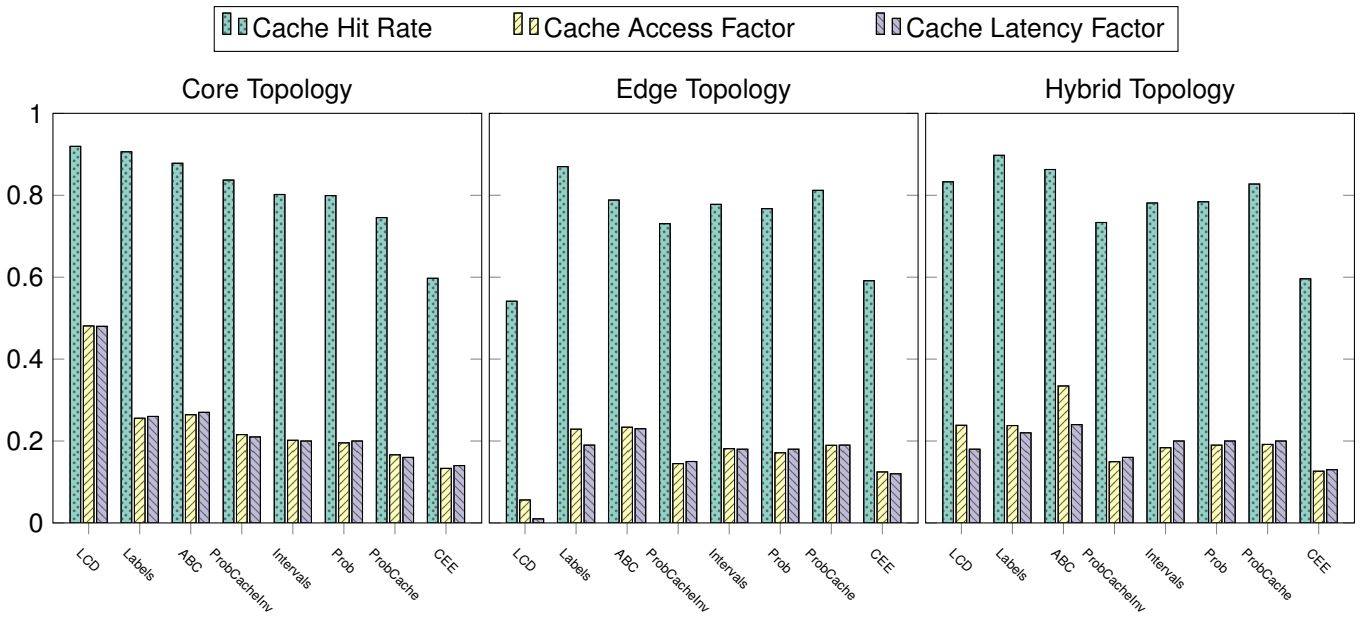



Fig. 6. Cache hit rate, cache access factor, and cache latency factor for different caching decision strategies

and satisfaction; cf: Section II-B2) is affected by the number of hops taken to retrieve the content. This does not differentiate between cached content and content produced by the prefix owner, i.e. the hop count shown here is the number of hops traversed by the Data packet to the requester from either its original producer or from a caching node. This figure essentially shows only the correlation between hop count and latency for individual transfers. As expected, this relation is indeed linear because nodes are evenly spaced in the network. The choice of caching strategy does not have a significant impact on this metric because, as will be shown below, the caching strategy affects the reduction in hop count over multiple hops and not the actual per-hop latency.

The fact that the relation between hop count and latency is linear underlines our claim that *the impact of cache misses is reduced in IoT*, as each additional hop only incurs a linear latency penalty. To show why this is an important observation, we will examine the cache hit rate next.

### B. Cache Hit Rate

Fig. 6 shows the cache hit rates for the different strategies, along with two more metrics that will be discussed later. Based on the cache hit rates, we could place the strategies into a tentative ordering that implies their relative performances. However, since the actual goal in most cases is to reduce content delivery latency, we should first examine how closely these two metrics are correlated.

### C. Hop Count Reduction

The top row of Fig. 7 shows the average hop count reduction for the different caching strategies at different distances. The first obvious effect is that most of the strategies only show a significant hop count reduction starting from a minimum distance to source. In all strategies except for *LCD*, there is a slight reduction at 3 hops and then a substantial one at 4 hops. The reason for this is that at shorter distances, there is less cache space between the producer and the consumer, which means fewer opportunities for content to be cached on the path. This makes it much more likely that a request will have to be routed all the way to the prefix owner to be satisfied. After a distance of 4 hops, the hops to hit will increase again as the distance to source increases. The "turning point" at which caching begins to have a noticeable impact seems to lie between 3 to 4 hops for most strategies. After this point, there is enough cache space on the path that content is likely to be found at a closer node.

The impact of the topology type can be most clearly seen when examining the performances of *ProbCache* and *ProbCacheInv* in the core and edge topologies. *ProbCache* [9] is a probabilistic strategy that favours caching content closer to the consumer (i.e. the edge). *ProbCacheInv* is our own variation on this strategy which simply inverts the probability so that content is more likely to be cached closer to the core instead. We can clearly see that depending on where paths intersect in the network, either one or the other strategy is clearly superior. In the hybrid topology, where path intersections are more evenly distributed, the two strategies are closer together in performance, with *ProbCache* narrowly beating out the inverted variant thanks to the high number of subtrees near the edge of the topology.

*LCD* is a bit of an outlier with extreme performance differences depending on topology. In the two extreme topology types it performs very strongly and very poorly respectively. This is due to the fact that *LCD* is very conservative and keeps contents very close to the core. This is beneficial in a topology where all paths intersect at the core, as it means that cached content can serve Interests from many sources, but entirely counterproductive if paths intersect at the edge of the network, as in this case Interests would still need to travel

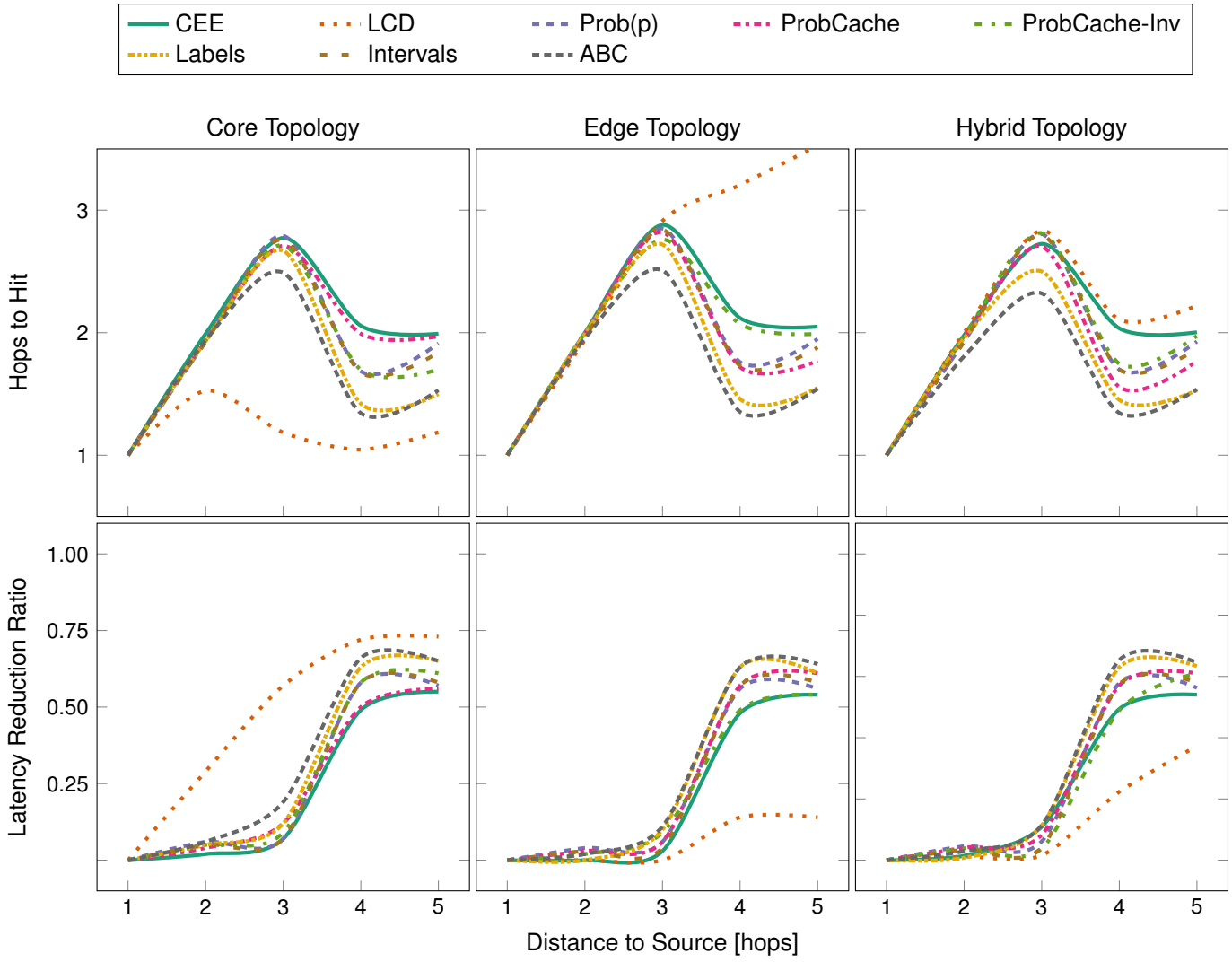



Fig. 7. Hop count, latency, and latency reduction for different caching reduction strategies

almost the entire distance to the core. In the more realistic hybrid topology, *LCD*'s behaviour is more similar to the other strategies, but its performance falls off especially at longer path lengths.

It has been observed multiple times [6], [9], [11], [51] that *CEE* is not an optimal caching strategy for ICN, and this is supported by the results presented here. The reason is that *CEE* is vulnerable to *thrashing* effects (especially if the *LRU* replacement policy is used, which is almost always the case) when nodes are caching high volumes of diverse data. The limited size of caches in IoT only exacerbates this effect.

### D. Latency Reduction Ratio

The results are shown in the bottom row of Fig. 7. The relative performances of the strategies are the same as for the $HRR$ in the top row, showing that hop reduction ratio is a clear indicator of expected latency reduction. However, if we contrast the results shown in Fig. 7 with the cache hit rate in Fig. 6, we can see some discrepancies in the relative performances that may have been inferred from the earlier figure. We will address these discrepancies and the shortcomings of the cache hit rate as a performance indicator in the following section.

### E. Demonstrating the Limits of the Cache Hit Rate

In Section III-A, we argued about limitations of cache hit rate in the context of IoT. We now demonstrate using our experimental results the lack of strong correlation between cache hit rate and content delivery latency. Fig. 8 shows the relation between these two metrics and the content delivery latency. We chose three strategies with distinct differences in $R_{CH}$ and plotted the relation between $R_{CH}$ and latency as well as $HRR_i$ and latency for 30 individual experiment runs. These results make it relatively obvious that the relation between cache hit rate and latency is not linear; simply increasing the hit rate does not guarantee a proportionate decrease in latency. While strategies with higher hit rates do have a very slight tendency towards lower latencies (since a cache is always closer to the consumer than the original producer), the difference is marginal. As mentioned above, this shows that even though a strategy may increase cache

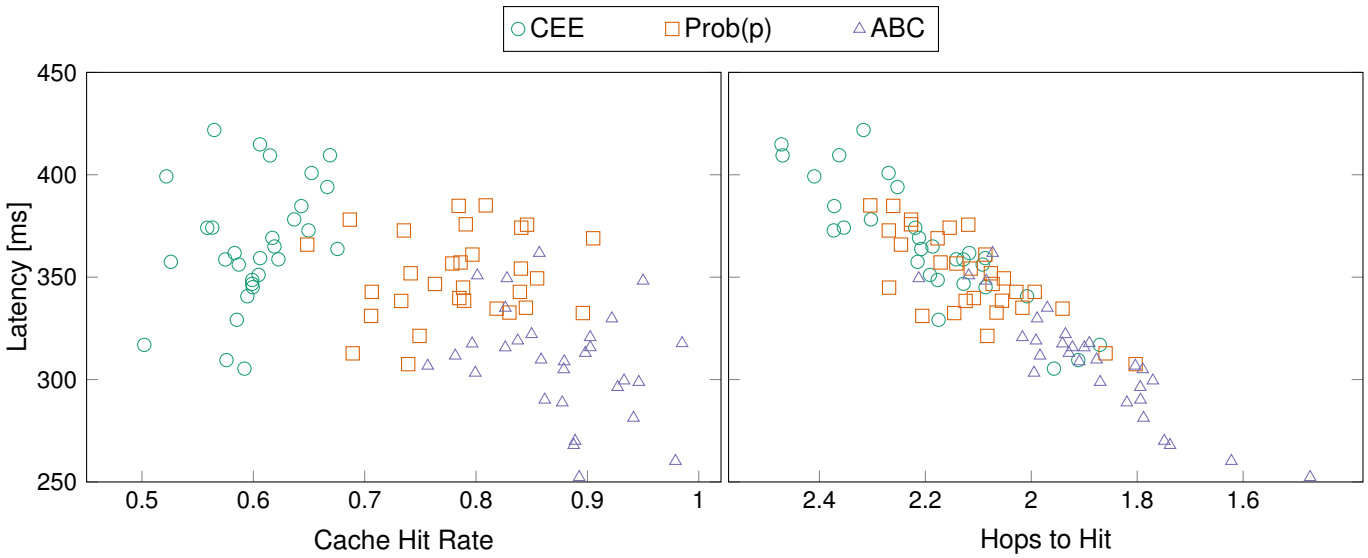



Fig. 8. Relation between cache hit rate, hops to hit, and content delivery latency for CEE, Prob(p), and ABC

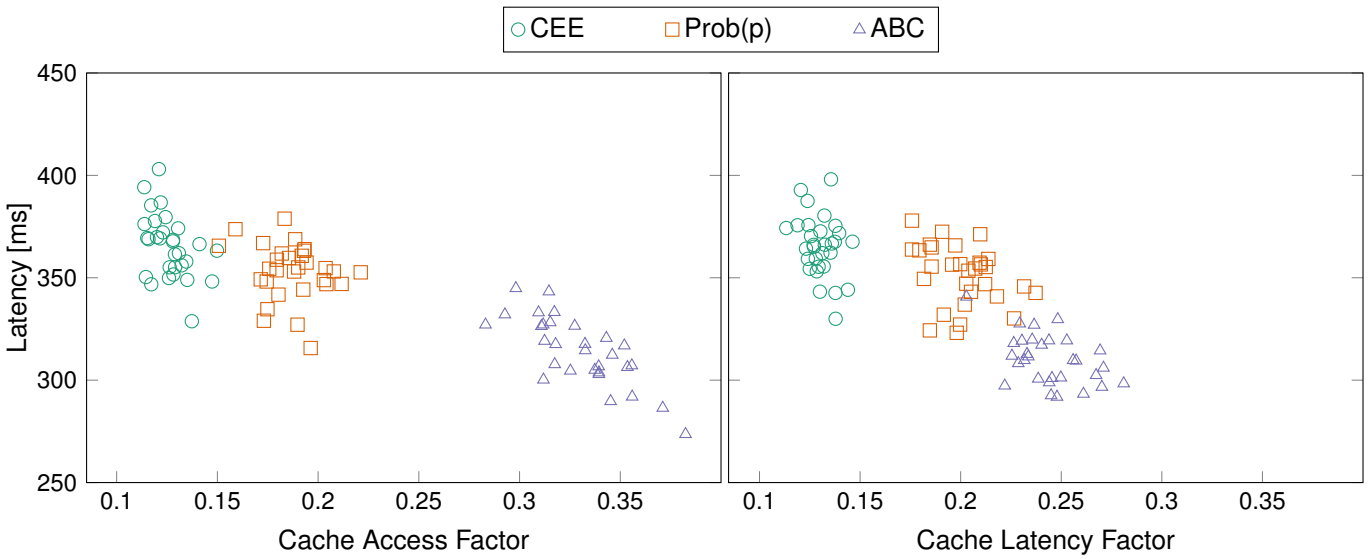



Fig. 9. Relation between cache access factor, cache latency factor, and content delivery latency for CEE, Prob(p), and ABC

hits, this alone does not improve performance if caches are not actually significantly closer to the consumer.

The reduction in hops to hit, on the other hand, exhibits a much more linear correlation with the reduction in latency. Consequently, this metric should have more impact on the choice of caching strategy in the IoT domain. It is further worth noting that while there is a strong stratification between strategies implied by the cache hit rate (the strategies occupy almost entirely separate sectors along the x axis on the left-hand side of Fig. 8), this stratification does not translate to the y axis, while in the right-hand plot, there is considerably more overlap between the strategies along the x axis, but a clear linear relation between x and y values.

### *F. Cache Access Factor and Cache Latency Factor*

The cache access and latency factor metrics are shown in relation to latency in Fig. 9, as well as in Fig. 6 along with the cache hit rate for comparison. Fig. 9 clearly shows that there is a much stronger correlation between both of these metrics and the latency than with the cache hit rate. While the average hops to hit as shown in Fig. 8 is the most strongly correlated with latency (which is to be expected), the new metrics are more nuanced and include more information while still indicating a general performance trend. Examining the new metrics in Fig. 6, we can see that while there is a slight overall trend of strategies doing well in terms of cache hit rate also doing well in the two new metrics, some pairwise comparisons using the new metrics actually produce results that disagree with the ordering implied by $R_{CH}$ and more closely align with the observations in Fig. 7. Therefore, we conclude that the cache access and cache latency factors provide valuable new insights into relative caching performance that would not have been possible using only the traditional metrics.

## VI. Conclusion

We have demonstrated how the traditional cache hit rate metric falls short of expectations when applied to the domain of information-centric IoT. We then introduced two new, comprehensive metrics, the *cache access factor* $F_{CA}$ and the *cache latency factor* $F_{CL}$, which provide a more nuanced understanding of a given strategy's expected performance. We showed the efficacy of these new metrics using experimental validation on a real IoT testbed provided by IoT-Lab.

While any performance evaluation should always strive to examine as many different metrics as feasible in order to paint a complete picture (and the cache hit rate absolutely has a place among these metrics), a metric that provides a simple way to predict a strategy's expected performance at a glance should prove very useful when presented with a wide variety of possible solutions.

## References


[1] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A Survey of Information-Centric Networking Research," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6563278/

[2] B. Liu, T. Jiang, Z. Wang, and Y. Cao, "Object-Oriented Network: A Named-Data Architecture Toward the Future Internet," *IEEE Internet of Things Journal*, vol. 4, no. 4, pp. 957–967, Aug. 2017.

[3] Z. Yan, S. Zeadally, and Y. Park, "A Novel Vehicular Information Network Architecture Based on Named Data Networking (NDN)," *IEEE Internet of Things Journal*, vol. 1, no. 6, pp. 525–532, Dec. 2014.

[4] S. Arshad, B. Shahzaad, M. A. Azam, J. Loo, S. H. Ahmed, and S. Aslam, "Hierarchical and Flat-Based Hybrid Naming Scheme in Content-Centric Networks of Things," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1070–1080, Apr. 2018.

[5] S. Arshad, M. A. Azam, S. H. Ahmed, and J. Loo, "Towards Information-Centric Networking (ICN) Naming for Internet of Things (IoT): The Case of Smart Campus," in *Proceedings of the International Conference on Future Networks and Distributed Systems*, ser. ICFNDS '17. New York, NY, USA: ACM, 2017, pp. 41:1–41:6, cambridge, United Kingdom. [Online]. Available: http://doi.acm.org/10.1145/3102304.3102345

[6] G. Carofiglio, V. Gehlen, and D. Perino, "Experimental Evaluation of Memory Management in Content-Centric Networking," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Kyoto, Japan, 5-9 Jun 2011, pp. 1–6.

[7] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache "Less for More" in Information-Centric Networks (Extended Version)," *Computer Communications*, vol. 36, no. 7, pp. 758–770, 2013.

[8] S.-E. Elayoubi and J. Roberts, "Performance and Cost Effectiveness of Caching in Mobile Access Networks," in *Proceedings of the 2nd ACM Conference on Information-Centric Networking*, ser. ACM-ICN '15. San Francisco, California, USA: Association for Computing Machinery, Sep. 2015, pp. 79–88. [Online]. Available: https://doi.org/10.1145/2810156.2810168

[9] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic In-Network Caching for Information-Centric Networks," in *Proceedings of the Second Edition of the ICN Workshop on Information-Centric Networking*, Helsinki, Finland, 17 Aug 2012, pp. 55–60.

[10] H. Zhu, Y. Cao, X. Wei, W. Wang, T. Jiang, and S. Jin, "Caching Transient Data for Internet of Things: A Deep Reinforcement Learning Approach," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2074–2083, Apr. 2019.

[11] J. Pfender, A. Valera, and W. K. G. Seah, "Performance Comparison of Caching Strategies for Information-Centric IoT," in *5th ACM Conference on Information-Centric Networking (ICN '18)*. Boston, MA, USA: ACM, 21-23 September 2018. [Online]. Available: http://conferences.sigcomm.org/acm-icn/2018/proceedings/icn18-final38.pdf

[12] G. Zhang, Y. Li, and T. Lin, "Caching in Information Centric Networking: A Survey," *Computer Networks*, vol. 57, no. 16, pp. 3128–3141, 2013.

[13] M. Zhang, H. Luo, and H. Zhang, "A Survey of Caching Mechanisms in Information-Centric Networking," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1473–1499, 2015.

[14] S. Tarnoi, K. Suksomboon, W. Kumwilaisak, and Y. Ji, "Performance of Probabilistic Caching and Cache Replacement Policies for Content-Centric Networks," in *Proceedings of the IEEE 39th Conference on Local Computer Networks (LCN)*, Edmonton, AB, Canada, 8-11 Oct 2014, pp. 99–106.

[15] I. U. Din, S. Hassan, M. K. Khan, M. Guizani, O. Ghazali, and A. Habbal, "Caching in Information-Centric Networking: Strategies, Challenges, and Future Research Directions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 2, pp. 1443–1474, 2018, conference Name: IEEE Communications Surveys Tutorials.

[16] C. V. Priscilla and A. Charulatha, "A Comparative Study on Caching Strategies in Content Centric Networking for Mobile Networks," in *2019 11th International Conference on Advanced Computing (ICoAC)*, Dec. 2019, pp. 122–128.

[17] M. A. Naeem, M. A. U. Rehman, R. Ullah, and B.-S. Kim, "A Comparative Performance Analysis of Popularity-Based Caching Strategies in Named Data Networking," *IEEE Access*, vol. 8, pp. 50 057–50 077, 2020.

[18] G. Carofiglio, L. Mekinda, and L. Muscariello, "FOCAL: Forwarding and Caching with Latency Awareness in Information-Centric Networking," in *2015 IEEE Globecom Workshops*, Dec. 2015, pp. 1–7.

[19] G. Carofiglio, L. Mekinda, and L. Muscariello, "LAC: Introducing Latency-Aware Caching in Information-Centric Networks," in *Proceedings of the IEEE 40th Conference on Local Computer Networks (LCN)*, Oct 2015, pp. 422–425.

[20] G. Carofiglio, L. Mekinda, and L. Muscariello, "Analysis of Latency-Aware Caching Strategies in Information-Centric Networking," in *Proceedings of the 1st Workshop on Content Caching and Delivery in Wireless Networks*, 2016, pp. 5:1–5:7.

[21] ——, "Joint Forwarding and Caching with Latency Awareness in Information-Centric Networking," *Computer Networks*, vol. 110, pp. 133–153, Dec. 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128616303176

[22] M. A. M. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "On the Performance of Caching and Forwarding in Information-Centric Networking for the IoT," in *Proceedings of the International Conference on Wired/Wireless Internet Communication (WWIC)*. Malaga, Spain: Springer, 25-27 May 2015, pp. 313–326.

[23] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, and K. Drira, "How to Cache in ICN-Based IoT Environments?" in *IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2017, pp. 1117–1124.

[24] S. Prybylski, M. Horowitz, and J. Hennessy, "Performance tradeoffs in cache design," in *Proceedings of the 15th Annual International Symposium on Computer Architecture*, ser. ISCA '88. Washington, DC, USA: IEEE Computer Society Press, 1988, pp. 290–298.

[25] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," in *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies*. ACM, 2009, pp. 1–12. [Online]. Available: http://dl.acm.org/citation.cfm?id=1658941

[26] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, and others, "Named Data Networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2656887

[27] S. Arshad, M. A. Azam, M. H. Rehmani, and J. Loo, "Information-Centric Networking Based Caching and Naming Schemes for Internet of Things: A Survey and Future Research Directions," *arXiv preprint arXiv:1710.03473*, 2017.

[28] L. Melvix, V. Lokesh, and G. C. Polyzos, "Energy Efficient Context Based Forwarding Strategy in Named Data Networking of Things," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*. ACM, 2016, pp. 249–254. [Online]. Available: http://dl.acm.org/citation.cfm?id=2988520

[29] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in Named Data Networking for the Wireless Internet of Things," in *International Conference on Recent Advances in Internet of Things (RIoT)*. IEEE, 2015, pp. 1–6.

[30] S. Arshad, M. A. Azam, M. H. Rehmani, and J. Loo, "Recent Advances in Information-Centric Networking-Based Internet of Things (ICN-IoT)," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2128–2158, Apr. 2019.

[31] D. Gupta, S. Rani, S. H. Ahmed, and R. Hussain, "Caching Policies in NDN-IoT Architecture," in *Integration of WSN and IoT for Smart Cities*, ser. EAI/Springer Innovations in Communication and Computing. Cham: Springer International Publishing, 2020, pp. 43–64. [Online]. Available: https://doi.org/10.1007/978-3-030-38516-3_3

[32] S. K. Fayazbakhsh, Y. Lin, A. Tootoonchian, A. Ghodsi, T. Koponen, B. Maggs, K. C. Ng, V. Sekar, and S. Shenker, "Less Pain, Most of the Gain: Incrementally Deployable ICN," in *ACM SIGCOMM Computer Communication Review*, vol. 43. ACM, 2013, pp. 147–158.

[33] J. Pfender, A. Valera, and W. K. G. Seah, "Content Delivery Latency of Caching Strategies for Information-Centric IoT," *arXiv:1905.01011 [cs]*, May 2019, arXiv: 1905.01011. [Online]. Available: http://arxiv.org/abs/1905.01011

[34] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and others, "FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed," in *Proceedings of the 2nd IEEE World Forum on Internet of Things (WF-IoT)*, Milan, Italy, 14-16 Dec 2015.

[35] Atmel, "AT86rf231 Low Power 2.4 GHz Transceiver for ZigBee, IEEE 802.15.4, 6LoWPAN, RF4CE, SP100, WirelessHART, and ISM Applications," 2009. [Online]. Available: http://www.atmel.com/images/doc8111.pdf

[36] "IEEE Standard for Low-Rate Wireless Networks," *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pp. 1–709, Apr. 2016.

[37] E. Baccelli, C. Gündoğan, O. Hahm, P. Kietzmann, M. S. Lenders, H. Petersen, K. Schleiser, T. C. Schmidt, and M. Wählisch, "RIOT: An Open Source Operating System for Low-End Embedded Devices in the IoT," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 4428–4440, Dec. 2018.

[38] E. Baccelli, O. Hahm, M. Güneş, M. Wählisch, and T. C. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2013, pp. 79–80. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6970748/

[39] C. Gündoğan, P. Kietzmann, M. Lenders, H. Petersen, T. C. Schmidt, and M. Wählisch, "NDN, CoAP, and MQTT: A Comparative Measurement Study in the IoT," in *Proceedings of the 5th ACM Conference on Information-Centric Networking*, ser. ICN '18. New York, NY, USA: ACM, 2018, pp. 159–171, boston, Massachusetts. [Online]. Available: http://doi.acm.org/10.1145/3267955.3267967

[40] C. Bormann, M. Ersue, and A. Keranen, "Terminology for Constrained-Node Networks," Internet Engineering Task Force (IETF), Tech. Rep., 2014.

[41] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information Centric Networking in the IoT: Experiments with NDN in the Wild," in *Proceedings of the 1st International Conference on Information-Centric Networking*. ACM, 2014, pp. 77–86. [Online]. Available: http://dl.acm.org/citation.cfm?id=2660144

[42] N. Laoutaris, H. Che, and I. Stavrakakis, "The LCD Interconnection of LRU Caches and its Analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, Jul. 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166531605000611

[43] J. Pfender, A. Valera, and W. K. G. Seah, "Easy as ABC: A Lightweight Centrality-Based Caching Strategy for Information-Centric IoT," in *Proceedings of the 6th ACM Conference on Information-Centric Networking*, ser. ICN '19. Macao, China: Association for Computing Machinery, Sep. 2019, pp. 100–111. [Online]. Available: https://doi.org/10.1145/3357150.3357405

[44] Z. Li and G. Simon, "Time-Shifted TV in Content Centric Networks: The Case for Cooperative In-Network Caching," in *Proceedings of the IEEE International Conference on Communications (ICC)*, Kyoto, Japan, 5-9 Jun 2011, pp. 1–6.

[45] Y. Zeng and X. Hong, "A Caching Strategy in Mobile Ad Hoc Named Data Network," in *6th International ICST Conference on Communications and Networking in China (CHINACOM)*. IEEE, 2011, pp. 805–809.

[46] H. Wang, J. M. Hernandez, and P. Van Mieghem, "Betweenness Centrality in a Weighted Network," *Physical Review E*, vol. 77, no. 4, p. 046105, 2008.

[47] L. Wang, S. Bayhan, and J. Kangasharju, "Effects of Cooperation Policy and Network Topology on Performance of In-Network Caching," *IEEE Communications Letters*, vol. 18, no. 4, pp. 680–683, Apr. 2014.

[48] M. Klymash, H. Beshley, O. Panchenko, and M. Beshley, "Method for optimal use of 4G/5G heterogeneous network resourses under M2M/IoT traffic growth conditions," in *Proceedings of the International Conference on Information and Telecommunication Technologies and Radio Electronics (UkrMiCo)*, Odessa, UKraine, 11-15 Sept 2017, pp. 1–5.

[49] W. Wang, C. Feng, B. Zhang, and H. Gao, "Environmental Monitoring Based on Fog Computing Paradigm and Internet of Things," *IEEE Access*, vol. 7, pp. 127 154–127 165, 2019.

[50] F. E. Ritter, M. J. Schoelles, K. S. Quigley, and L. C. Klein, "Determining the number of model runs: Treating cognitive models as theories by not sampling their behavior," in *Human-in-the-loop simulations: Methods and practice*, S. Narayanan and L. Rothrock, Eds. Springer-Verlag, 2011, pp. 97–116.

[51] I. Psaras, R. G. Clegg, R. Landa, W. K. Chai, and G. Pavlou, "Modelling and Evaluation of CCN-Caching Trees," in *Proceedings of the 10th International IFIP TC 6 Conference on Networking*, ser. NETWORKING, Valencia, Spain, 9-13 May 2011, pp. 78–91. [Online]. Available: http://dl.acm.org/citation.cfm?id=2008780.2008789


## List of Acronyms

ABC Approximate Betweenness Centrality.

CCN Content-Centric Networking.
CEE Cache Everything Everywhere.
CS Content Store.

FIB Forwarding Information Base.

ICN Information-Centric Networking.
IoT Internet of Things.

LCD Leave Copy Down.
LCE Leave Copy Everywhere.
LRU Least Recently Used.

NDN Named Data Networking.

PIT Pending Interest Table.

WSN Wireless Sensor Network.